

# REPORT DOCUMENTATION PAGE

AFRL-SR-AR-TR-03-

the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or reducing this burden to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503

0320

aining  
s for  
ce of

1. AGENCY USE ONLY (Leave blank) 2. REPORT DATE 08/12/2003 3. REPORT TYPE AND DATES COVERED Final Performance Report 01/01/2000-05/30/2003

4. TITLE AND SUBTITLE  
Interactive Anticipatory Scheduling for Two Military Applications

5. FUNDING NUMBERS  
AFOSR #F49620-00-1-0144

6. AUTHOR(S)  
Adele Howe and L. Darrell Whitley

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)  
Colorado State University  
Fort Collins, CO 80523

8. PERFORMING ORGANIZATION  
REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)  
Air Force Office of Scientific Research  
4015 Wilson Blvd, Room 713  
Arlington, VA 22203-1954

10. SPONSORING / MONITORING  
AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

20030915 034

12a. DISTRIBUTION / AVAILABILITY STATEMENT  
Unclassified - Approved for public release

Approved for public release,  
distribution unlimited

12b. DISTRIBUTION CODE  
A

13. ABSTRACT (Maximum 200 Words)  
Application of scheduling technologies lags significantly behind the state-of-the-art. This project investigated two contributors to this lag. First, researchers do not know what makes particular problems difficult for their methods. Second, researchers often develop methods in isolation from actual data and applications. To address the first, new static and the first dynamic models of local search algorithms have been developed; these models partially explain what makes some job shop scheduling problems difficult. For the second, several algorithms for Air Force Satellite Control Network scheduling have been compared on historical and recent data. Additionally, a prototype interactive scheduler has been built that includes the algorithm and objective functions.

14. SUBJECT TERMS  
operations research, artificial intelligence, scheduling

15. NUMBER OF PAGES  
33

16. PRICE CODE

17. SECURITY CLASSIFICATION  
OF REPORT

18. SECURITY CLASSIFICATION  
OF THIS PAGE

19. SECURITY CLASSIFICATION  
OF ABSTRACT

20. LIMITATION OF ABSTRACT

Final Report for AFOSR #F49620-00-1-0144  
*Interactive Anticipatory Scheduling for Two Military  
Applications*

January 1, 2000 to May 30, 2003

Adele Howe    L. Darrell Whitley  
Computer Science Department  
Colorado State University  
Fort Collins, CO 80524  
email: {howe,whitley}@cs.colostate.edu

August 12, 2003

**Abstract**

Application of scheduling technologies lag significantly behind the state of the art. This project investigated two contributors to this lag. First, researchers do not know what makes particular problems difficult for their methods. Second, researchers often develop methods in isolation from actual data and applications. To address the first, new static and the first dynamic models of local search algorithms have been developed; these models partially explain what makes some job shop scheduling problems difficult. For the second, several algorithms for Air Force Satellite Control Network scheduling have been compared on historical and recent data. Additionally, a prototype interactive scheduler has been built that includes the algorithms and alternative objective functions.

## Contents

<b>1</b>	<b>Project Objectives</b>	<b>3</b>
<b>2</b>	<b>Job Shop Scheduling Modeling and Studies</b>	<b>4</b>
2.1	Evaluation Methodology . . . . .	4
2.2	Static Models . . . . .	5
2.3	Dynamic Models . . . . .	10
<b>3</b>	<b>Air Force Satellite Control Network (AFSCN) Scheduling</b>	<b>13</b>
3.1	Variants of Range Scheduling with Differing Complexity . . . . .	15
3.2	Algorithms for the Satellite Range Scheduling Problem . . . . .	16
3.3	The Problem Generator . . . . .	18
3.4	Multi-Resource Range Scheduling with Alternatives: Algorithm Analysis and the AFIT Benchmark Problems . . . . .	19
3.4.1	Explaining the Performance on the Benchmarks . . . . .	20
3.5	Generalizing the Satellite Scheduling Results: A Preliminary Study . . . . .	21
3.6	Java Interface to AFSCN Algorithms . . . . .	22
3.7	Conclusions about AFSCN Range Scheduling . . . . .	23
<b>4</b>	<b>Accomplishments/New Findings</b>	<b>24</b>
4.1	Job Shop Scheduling Modeling and Studies . . . . .	24
4.2	Air Force Satellite Scheduling . . . . .	25
<b>5</b>	<b>Executive Summary</b>	<b>26</b>
5.1	Personnel . . . . .	26
5.2	Publications . . . . .	26
5.3	Graduate Theses . . . . .	27
5.4	Other Products . . . . .	29

# 1 Project Objectives

Air Force Satellite Control Network (AFSCN) scheduling is a time consuming process requiring a staff of experienced human schedulers. A single day's schedule may take a week of work for the human scheduler. A compelling case can be made that automation can reduce the scheduling time window and provide tools for human schedulers to considerably improve the efficiency of the resulting schedules. However, efforts to automate even small parts of the process have had little success. Implementation of automated methods lags far behind the state of the art in this application.

The primary objectives of this project were to investigate two factors that contribute to this lag in deploying scheduling methods:

1. a lack of scientific understanding of what makes some scheduling problems difficult for some state-of-the-art methods, and
2. a disparity between the data/domains being used by researchers for algorithm development/testing and the data/domains for actual deployed systems.

For both of these, state of the art algorithms were implemented and empirically evaluated on a variety of problems: benchmark, artificially generated and actual data.

To better understand search algorithm performance, a set of models that relates search space features to search cost were developed. These models are intended to focus attention on the most difficult problems, to identify algorithm characteristics that are well suited to such problems, and to explain why algorithms perform well when they do. The models are intended to be used to motivate improvements to existing algorithms and motivate selection of particular algorithms for particular problems.

For this part of the project, Job Shop Scheduling (JSP) was studied because many algorithms have been developed for it and many observations have been published regarding relative algorithm performance and relative problem difficulty. Our performance models can be used to explain why certain JSP problems are more difficult and to explain underlying causes of some published observations.

To address the second objective (the disparity between research and practice), the AFSCN application was studied. Historical and recent data was obtained to assess how the problem has changed over time. Problem generators were constructed based on actual data that allow studies controlling varying problem characteristics. Several algorithms were implemented, which were thought to be appropriate for solving this application, and their performance was studied on different problems.

Finally, a prototype interactive scheduling system for AFSCN scheduling was constructed. Using it, the human scheduler is able to interactively select or reject parts of the schedule and fix these decisions. The automated system updates, repairs and improves schedules in response to changes made by the human. The support environment also provides capability of switching schedule evaluation criteria and algorithms used. Proposed schedules are displayed so that remaining conflicts can be visualized and alternatives examined. The system is intended to showcase how the AFSCN human schedulers could be supported and to comparatively evaluate alternative approaches to partially automating this scheduling.

The remainder of this report is divided into two parts related to these factors. The results of our work have appeared in the publications listed in section 5.2; the next two sections summarize the underlying efforts for the two parts. Section 4 enumerates the primary accomplishments for these two factors.

## 2 Job Shop Scheduling Modeling and Studies

Job Shop scheduling is perhaps the most studied scheduling application. Over the years, a wide range of algorithms have been developed for it with attention shifting based on which algorithm class seemed to be state of the art at any given time. The most consistent class of performers has been local search (e.g., simulated annealing, iterated local search and tabu search), with the variants becoming increasingly complicated to obtain superior performance. Unfortunately, the designs of the algorithms were based largely on experience in working with the problem and not on a deep model of problem complexity and algorithm performance.

A series of experiments was conducted to model local search algorithms. The aim was to understand why this class of algorithm appears to excel and to determine what features most contribute to superior performance. Once these issues were understood, then new principled algorithms for the problem could be designed.

This section briefly describes the project's research on evaluation methodology, the models that were developed for job shop scheduling and the experiments that support them. Full details of the experiments, models and algorithms can be found in Jean-Paul Watson's 2003 Ph.D. thesis [35]; several parts have been separately published as referenced in this section.

### 2.1 Evaluation Methodology

Scheduling algorithms typically are evaluated by comparing their performance to that of some well known alternatives on well known, difficult benchmark problems. Although it is standard operating procedure, such methodology limits what can be reasonably inferred about the performance of the new algorithm and its suitability for particular applications and even specific problems within applications.

This methodology was studied to better understand its limitations. For example, although benchmark problems provide an objective means of comparing systems, systems can become overfitted to work well on benchmarks and, therefore, that good performance on benchmarks does not generalize to real-world problems [45].

Our study showed that definitions of problem difficulty are strongly biased ("it is difficult if the state-of-the-art algorithm cannot easily solve it") and are focused on trends rather than individual variation [39]. "Difficult" problems also may be unrealistic. The best known job shop scheduling benchmark collection (O.R. library[5]) was produced by generating problems randomly and then filtering them for problem difficulty. The more difficult random problems tend to have less structure that could be exploited by heuristics; in effect, the random problems could be harder than the actual applications [36].

The effects of assumptions underlying comparative evaluation in a related area, planning, were studied as well [18]. Because the planning community sponsors a biennial competition, the methodology has been debated and refined publicly; additionally, the systems have been

made public<sup>1</sup>. These factors simplified a rigorous study of the methodology. Our study examined the impact of major experiment decisions, e.g., software versions, problem selection, minor variations on problems, time cut-offs, and hardware differences, on the outcome of comparative experiments. Thirteen planners were run on approximately 500 benchmark problems and another 1000 permutations of some of them under varying hardware and software environments. A high level of sensitivity was found to many of the factors, especially those involving the selection of problems to solve, but surprisingly less sensitivity to time cut-offs and hardware differences.

## 2.2 Static Models

Local search algorithms are among the most effective approaches for solving the JSP, yet there is little understanding of why these algorithms work so well, and under what conditions. Our project developed static models of problem difficulty for local search for the job shop scheduling problem [37,38,40]. Static models relate features of search space to search cost; they are static because they are a post-hoc analysis of the result of search. To the best of our knowledge, we are the only researchers who have published such models. Further, only one group of researchers, Mattfeld et al. [20], has analyzed the link between problem difficulty and local search for the JSP in general.

Problem difficulty has been modeled for the Boolean Satisfiability Problem (SAT). Recent models account for much of the variance in local search cost observed in a particular class of random problem instances commonly known as Random 3-SAT [10,23,25]. All of these models are based on particular features of the search space and on the assumption that a particular feature, or set of features, is largely responsible for the cost required by a local search algorithm to locate an optimal solution to a problem instance. The assumption is generally tested via linear or multiple regression methods, with the regression  $r^2$  value quantifying the accuracy of the resulting model.

The models developed for SAT suggested a set of search space features:

- number of optimal solutions,
- backbone size (number of Boolean variables that have the same truth value in all optimal solutions),
- the distance between initial solutions and the nearest optimal solution, and
- backbone robustness (number of clauses that can be deleted before the backbone size is reduced by at least half).

These features were translated to JSP. The models were constructed by solving  $6 \times 4$  and  $6 \times 6$  JSP problems (both benchmarks and generated instances) and calculating the above search space features as well as search cost (how many iterations are required to locate an optimal solution). Three of the four cost models require computation of *all* optimal solutions to a

<sup>1</sup>This level of uniformity and accessibility is not evident in the scheduling community yet, but a competition series has recently begun for scheduling sponsored by the French Operations Research Society.

problem instance, which can number in the tens of millions for these problem instances. Search cost was defined as the median search cost over 5000 independent runs of the algorithms.

The static model characterizes the topology of the search space that local search algorithms traverse. The local search algorithm was the tabu search algorithm introduced by Taillard [31], which is denoted by  $TS_{Taillard}$ , because its core was common to state-of-the-art tabu search for the JSP, but it was simpler and so easier to analyze.

**Number of Optimal Solutions** Intuitively, a decrease in the number of optimal solutions (denoted  $|optsols|$ ) should yield an increase in local search cost. Figure 1 shows scatter-plots of  $\log_{10}(|optsols|)$  versus  $\log_{10}(cost_{med})$  for  $6 \times 4$  and  $6 \times 6$  general JSPs. The  $r^2$  values for the corresponding regression models are 0.5365 and 0.2223, respectively.

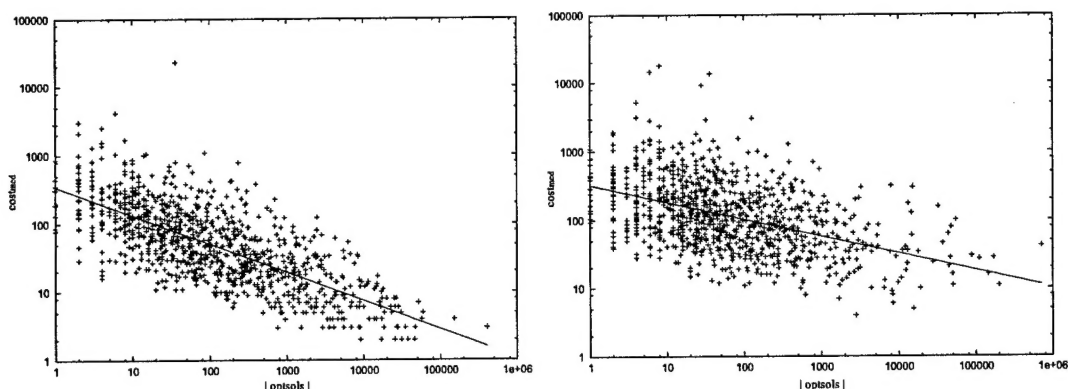


Figure 1: Scatter plots of  $\log_{10}(|optsols|)$  versus  $\log_{10}(cost_{med})$  for  $6 \times 4$  (left figure) and  $6 \times 6$  (right figure) general JSPs; the least-squares fit lines are superimposed. The  $r^2$  values for the corresponding regression models are 0.5307 and 0.2231, respectively.

The results presented in Figure 1 indicate that for typical general JSPs, a descriptive cost model based on  $|optsols|$  is relatively inaccurate, accounting for roughly 50% of the variance in search cost in the *best* case. In the general JSP, as  $n/m \rightarrow \infty$ , the frequency of problem instances with a large number of optimal solutions increases. By extrapolation, one would then expect the accuracy of the  $|optsols|$  model to increase as  $n/m \rightarrow \infty$ . In contrast, the accuracy of the model appears worst for the most difficult class of general JSP (i.e., those with  $n/m \approx 1.0$ ), with model residuals varying over 2 to 3 orders of magnitude.

**Backbone Size** The definition of a backbone clearly depends on how solutions are represented. The most common solution encoding used in local search algorithms for the JSP, including  $TS_{Taillard}$ , is the *disjunctive graph* [7]. In the disjunctive graph representation, there are  $n(n-1)/2$  Boolean ‘order’ variables for each of the  $m$  machines, each of which represents a precedence relation between a distinct pair of jobs on a machine. Consequently, the backbone of a JSP is defined as the set of Boolean order variables that have the same value in all optimal solutions; the backbone *size* is defined as the fraction of the possible  $mn(n-1)/2$  order variables that are fixed to the same value in all optimal solutions, which is denoted by  $|backbone|$ .

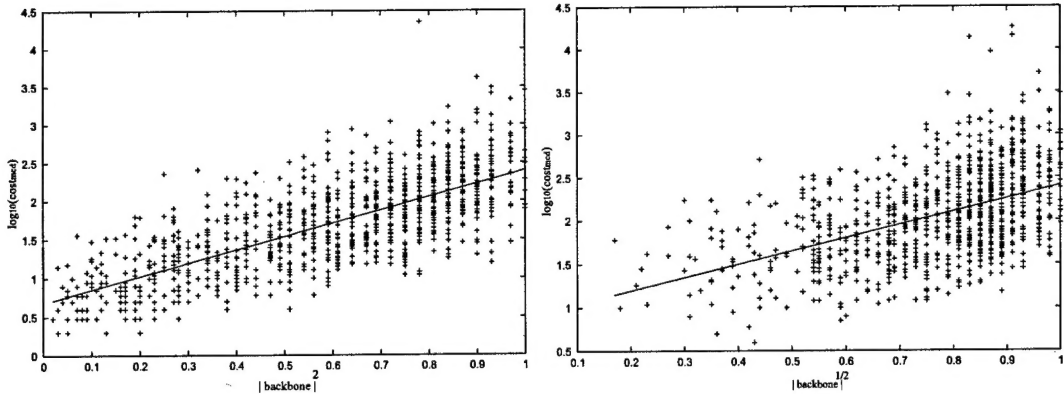


Figure 2: Scatter plots of  $|backbone|^2$  versus  $\log_{10}(cost_{med})$  for  $6 \times 4$  (left figure) and  $6 \times 6$  (right figure) general JSPs; the least-squares fit lines are superimposed. The  $r^2$  values for the corresponding regression models are 0.5307 and 0.2231, respectively.

Figure 2 shows scatter plots of  $|backbone|^2$  versus  $\log_{10}(cost_{med})$  for  $6 \times 4$  and  $6 \times 6$  general JSPs. The  $r^2$  values for the corresponding regression models are 0.5307 and 0.2331, respectively.

The data show a close correspondence between the  $r^2$  values of the  $|backbone|$  and  $|optsols|$  models. This phenomenon appears to be due to an extremely high correlation between  $|backbone|^2$  and  $\log_{10}(|optsols|)$ :  $-0.9337$  and  $-0.9103$  for  $6 \times 4$  and  $6 \times 6$  problems, respectively. Within each problem group, the correlation is near perfect for instances with large backbones and gradually decays as  $|backbone| \rightarrow 0.0$ . Our results indicate that, somewhat surprisingly, for problem instances with moderate-to-large backbones, the backbone size is essentially a proxy for the number of optimal solutions, and vice-versa. From the standpoint of models of problem difficulty for reasonably difficult general JSPs (i.e., those with moderate-to-large backbones), the two features are redundant.

**Average Distance Between Random Local Optima** Search in algorithms with a strong bias toward local optima is largely constrained to the subspace of local optima. Consequently, one would expect search cost in these algorithms to be at least somewhat correlated with the size of this subspace. For each of our general JSPs, the average normalized distance between distinct random pairs of local optima ( $\overline{loptdist}$ ) was computed using a set of 5000 random local optima produced using a steepest-descent procedure. Scatter plots of  $\overline{loptdist}$  versus  $\log_{10}(cost_{med})$  for  $6 \times 4$  and  $6 \times 6$  general JSPs are shown in Figure 3. The  $r^2$  values for the corresponding regression models are 0.2415 and 0.2744, respectively. These results confirm the intuition that the size of the local optima subspace is correlated with the cost of finding optimal solutions under  $TS_{Taillard}$ , albeit more weakly than either  $|optsols|$  or  $|backbone|$  in  $6 \times 4$  general JSPs (i.e.,  $r^2$  values of 0.2415 versus 0.5365 and 0.5307, respectively). In contrast to both  $|optsols|$  and  $|backbone|$ , the strength of the  $\overline{loptdist}$  model is largely insensitive to relatively small changes in the problem dimensions.

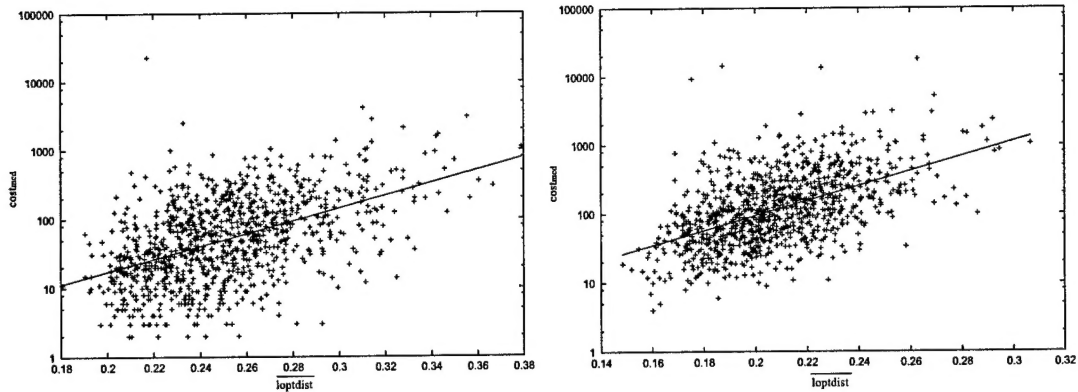


Figure 3: Scatter plots of  $\log_{10}(\overline{loptdist})$  versus  $cost_{med}$  for  $6 \times 4$  (left figure) and  $6 \times 6$  (right figure) general JSPs; the least-squares fit lines are superimposed. The  $r^2$  values for the corresponding regression models are 0.2415 and 0.2744, respectively.

**Distance between Random Local Optima and the Nearest Optimal Solution** Local search algorithms for SAT quickly locate suboptimal *quasi-solutions* in which relatively few clauses are unsatisfied. These quasi-solutions form a subspace that contains all optimal solutions; once a solution in this subspace is identified, local search is typically restricted to this subspace. An obvious analog of quasi-solutions in SAT are local optima in the JSP. Another static model is based on the distance from members of this subspace to the nearest optimum.

To produce this model, 5000 random local optima were generated for each problem using a steepest-descent procedure. The mean normalized distance between the resulting local optima and the nearest optimal solution was computed (denoted by  $d_{lopt-opt}$ ). The distances are normalized to enable comparisons between  $6 \times 4$  and  $6 \times 6$  general JSPs and are scaled by square root to accommodate a slight curvature in the residual plots for small values of  $d_{lopt-opt}$ . scatter plots of  $\sqrt{d_{lopt-opt}}$  versus  $\log_{10}(cost_{med})$  for  $6 \times 4$  and  $6 \times 6$  general JSPs are shown in Figure 4.

Clearly, the  $d_{lopt-opt}$  model is significantly more accurate than any of the  $|optsols|$ ,  $|backbone|$ , or  $\overline{loptdist}$  models. This model accounts for a substantial proportion of the variance in the cost of  $TS_{Taillard}$  in typical general JSP.

**Implications and Limitations of the Best Static Model:  $d_{lopt-opt}$**  Empirical evidence showed that square JSPs (number of machines  $m$  approaches the number of jobs  $n$  or  $\frac{m}{n} = 1$ ) were generally more difficult than rectangular; unfortunately, no one knew why. The  $d_{lopt-opt}$  model can serve as a partial explanation for that phenomenon.

To model the differences, 10 000 general JSPs were generated for  $m = 3$  and  $n = 4$  through  $n = 7$ . Histograms of  $d_{lopt-opt}$  for  $4 \times 3$  and  $7 \times 3$  are shown in Figure 5. In  $4 \times 3$  general JSPs, the right-tail mass of the distribution is substantial (e.g., for  $d_{lopt-opt} \geq 0.3$ ), especially in comparison to the distribution for  $7 \times 3$  general JSPs, where instances with  $d_{lopt-opt} \geq 0.3$  are quite rare. Similar histograms for general JSPs with  $n/m < 1$  continue the distribution mass shift toward 0.5. Although not entirely conclusive, our results provide relatively strong

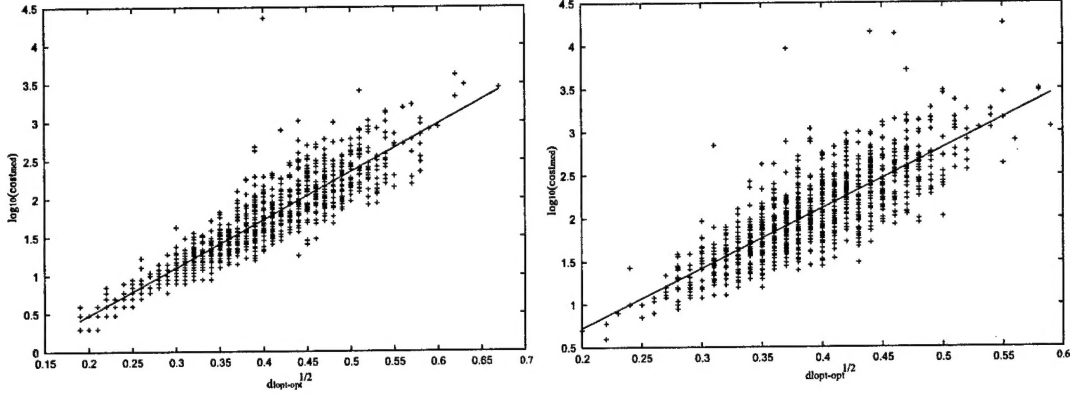


Figure 4: scatter plots of  $\sqrt{d_{lopt-opt}}$  versus  $\log_{10}(cost_{med})$  for  $6 \times 4$  (left figure) and  $6 \times 6$  (right figure) general JSPs; the least-squares fit lines are superimposed. The  $r^2$  values for the corresponding regression models are 0.826 and 0.6541, respectively.

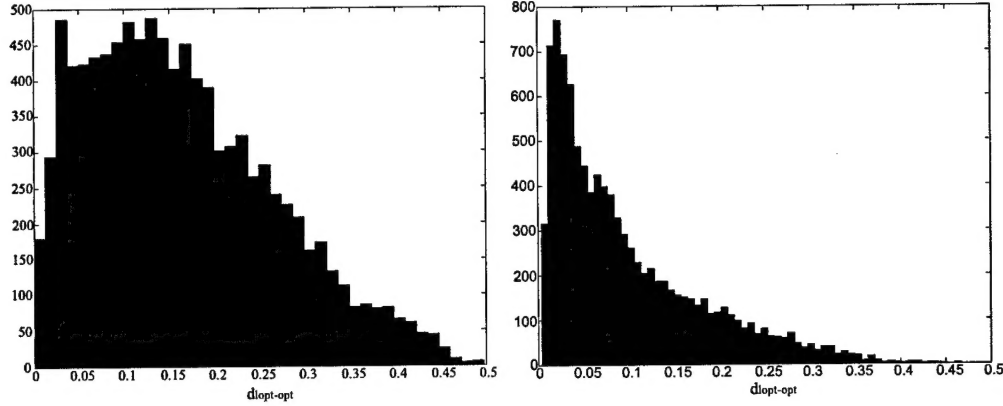


Figure 5: Histograms of  $d_{lopt-opt}$  for 10 000  $4 \times 3$  (left figure) and  $7 \times 3$  (right figure) general JSPs.

evidence that the right-tail mass of the  $d_{lopt-opt}$  distribution vanishes as  $n/m \rightarrow \infty$ .

The  $d_{lopt-opt}$  model has several known limitations. First, the model was developed using small problem instances, because of the need to exhaustively enumerate local optima. Second, it is less accurate for problem instances with large values of  $d_{lopt-opt}$  (or, equivalently, large  $cost_{med}$ ). Third, the accuracy of the  $d_{lopt-opt}$  model is exceptionally poor for very high-cost, general JSPs.

Finally, the  $d_{lopt-opt}$  model is unable to account for a significant proportion of the variance in the cost of finding optimal solutions of more structured JSPs: e.g., workflow JSPs. Our models of the different types of JSPs exhibited significant differences in accuracy, which provides even more evidence that generalizing results based on random problems is risky.

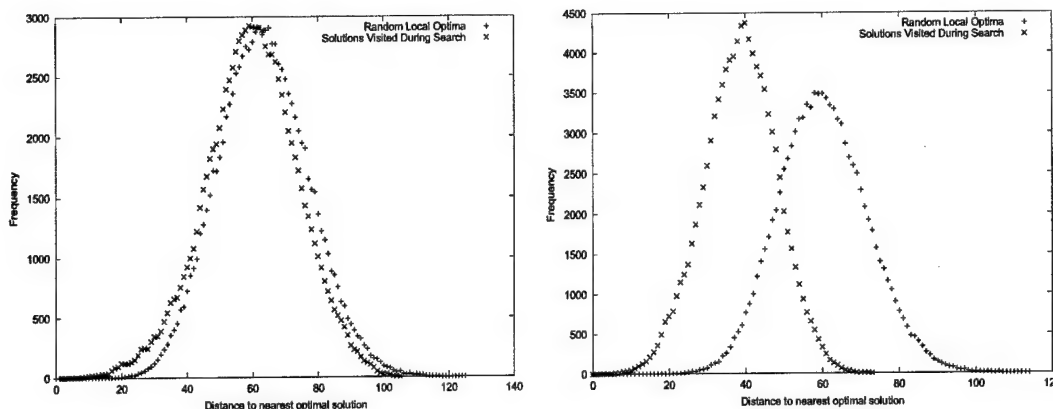


Figure 6: Histograms of the distance to the nearest optimal solution ( $d_{opt}$ ) for (a) 100,000 random local optima and (b) 100,000 solutions generated by  $TS_{Taillard}$  for two  $10 \times 10$  random JSPs.

### 2.3 Dynamic Models

Static cost models adopt a retrospective view of search; one can, at best, indirectly infer what might have happened during the search process. In contrast, dynamic models are high-resolution models of the run-time behavior of search algorithms; these models help to explain *why* particular features are correlated with search cost. Consequently, developing dynamic models appears to be the best way to address the limitations of the static models.

**Quasi-Dynamic Models** are based on aggregate statistics of run-time behavior. The  $\bar{d}_{lopt-opt}$  measure is a function of the distribution of the distance between *random* local optima and the nearest optimal solution ( $d_{opt}$ ). However, as shown in Figure 6, the distribution of  $d_{opt}$  for solutions visited by  $TS_{Taillard}$  during search (right side of figure) differs in both means and variances from the distribution for random local optima. This phenomenon becomes more pronounced with larger problems (i.e., distributions are identical for  $6 \times 4$  instances and start to diverge for  $6 \times 6$ ), mirroring the reduction in accuracy of the  $d_{lopt-opt}$  model.

Our quasi-dynamic models [41] replace the mean  $d_{opt}$  for solutions visited during search (denoted  $\bar{d}_{tabu-opt}$ ) for  $\bar{d}_{lopt-opt}$ . For a given instance,  $\bar{d}_{tabu-opt}$  is computed using a set of 100,000 solutions visited by  $TS_{Taillard}$  over a variable number of independent trials. Each trial is initiated from a random local optimum and terminated once an optimal solution is located.

Regression models of  $\bar{d}_{tabu-opt}$  versus  $\log_{10}(c_{Q2})$  yielded  $r^2$  values of 0.8441 for our  $6 \times 4$  instances and 0.7808 for our  $6 \times 6$  instances; this corresponds to roughly 4% and 20% increases in accuracy over that of the  $\bar{d}_{lopt-opt}$  model, respectively. The scatter-plot for the  $6 \times 6$  instances is shown in the left side of Figure 7. The right side of Figure 7 shows the results on a set of forty-two computationally tractable  $10 \times 10$  instances; these instances tended to be problematic for our static models. The regression model of  $\bar{d}_{tabu-opt}$  versus  $\log_{10}(c_{Q2})$  for these  $10 \times 10$  instances yielded an  $r^2$  value of 0.6641, a 41% increase in accuracy over the static model. The scatter-plot is annotated with data for the five tractable instances of the seven  $10 \times 10$  random

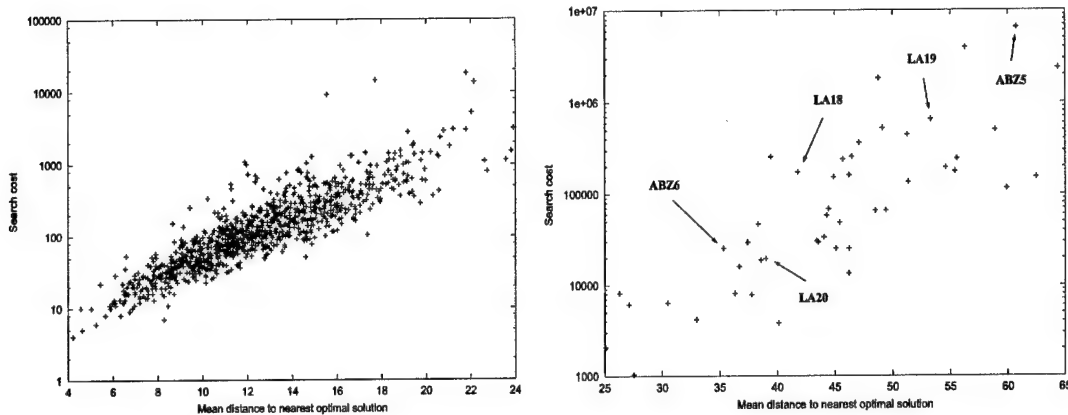


Figure 7: scatter plots of  $\bar{d}_{\text{tabu-opt}}$  versus search cost ( $c_{Q2}$ ) for  $6 \times 6$  (left figure) and  $10 \times 10$  (right figure) random JSPs; the least-squares fit lines are superimposed.

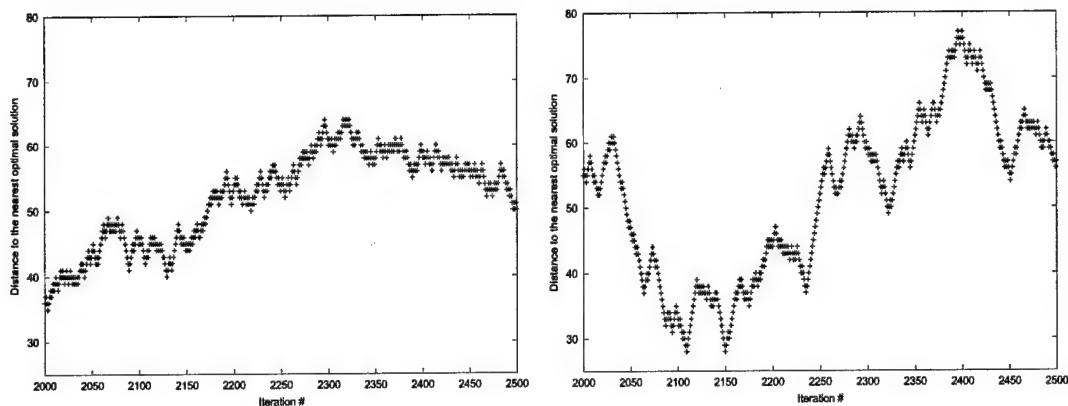


Figure 8: Time series of the distance to the nearest optimal solution for the solutions visited by a random walk (left figure) and  $TS_{\text{Taillard}}$  (right figure) for a typical  $10 \times 10$  random JSP.

JSPs found in the OR Library.

**Dynamic Cost Model for Tabu Search** Our dynamic cost model for tabu search [41] is a Markov chain in which each state  $S_{i,grad}$  is defined as a pair of (1) the distance  $i$  to the nearest optimal solution and (2) the search gradient  $grad$  toward (*closer*), at the same distance (*equal*) or away (*farther*) from the nearest optimal solution. The addition of search gradient information was intended to model the impact of short-term memory. Tabu search appears to exhibit such trending behavior; Figure 8 shows a time series of the distance to the nearest optimal solution for a random walk (left figure) and  $TS_{\text{Taillard}}$  (right figure) for a  $10 \times 10$  random JSP. Tabu search maintains a direction for considerably longer than the random walk. Given a maximum possible distance of  $D$  from a solution to the nearest optimal solution, our Markov model consists of exactly  $3 \cdot (D + 1)$  states (the “+1” state represents the set of optimal solutions).

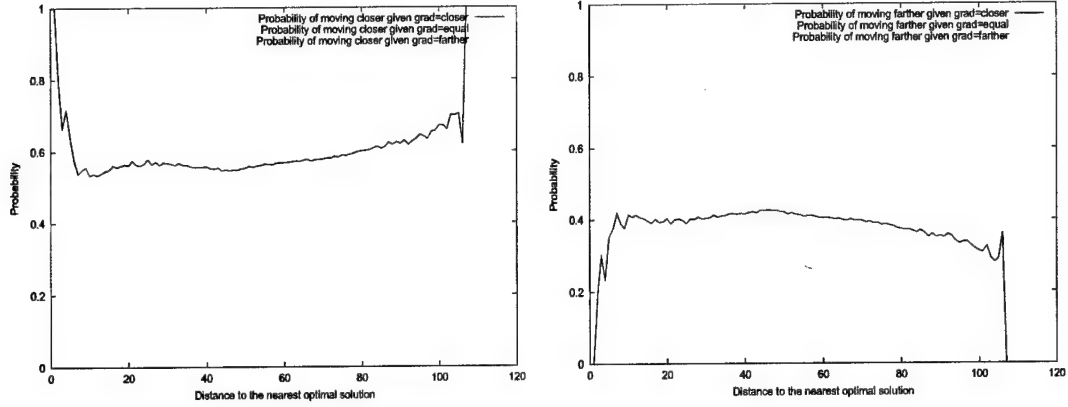


Figure 9: The transition probabilities for moving closer to (left figure) or farther from (right figure) the nearest optimal solution under  $TS_{Taillard}$  for a typical  $10 \times 10$  random JSP.

The transition probabilities between pairs of states  $S_{i,grad}$  and  $S_{j,grad}$  are the conditional probabilities  $P(S_{i,grad}|S_{j,grad})$ . The majority of these probabilities obviously equal 0: specifically, for any pair of states  $S_{i,grad}$  and  $S_{j,grad}$  with  $|i - j| > 1$ , or when simultaneous changes in both gradient and distance to the nearest optimal solution are logically impossible, such as from state  $S_{i,closer}$  to state  $S_{i+1,closer}$ . The set of transition probabilities is also subject to the total-probability constraints:

- $P(S_{i-1,closer}|S_{i,closer}) + P(S_{i,equal}|S_{i,closer}) + P(S_{i+1,farther}|S_{i,closer}) = 1.0$
- $P(S_{i-1,closer}|S_{i,equal}) + P(S_{i,equal}|S_{i,equal}) + P(S_{i+1,farther}|S_{i,equal}) = 1.0$
- $P(S_{i-1,closer}|S_{i,farther}) + P(S_{i,equal}|S_{i,farther}) + P(S_{i+1,farther}|S_{i,farther}) = 1.0$

The Markov model includes a reflective barrier at  $i = D$  and an absorbing state at  $i = 0$  by imposing the constraints  $P(S_{D+1,farther}|S_{D,farther}) = 0$  and  $P(S_{0,equal}|S_{0,equal}) = 1$ , respectively. We estimate the set of transition probabilities for a given problem instance by analyzing the set of solutions visited by  $TS_{Taillard}$  over a large number of independent trials.

Figure 9 shows the estimated probabilities of moving closer to (left figure) or farther from (right figure) the nearest optimal solution for a typical  $10 \times 10$  random JSP; the probability of maintaining an *equal* search gradient is negligible ( $p < 0.1$ ) for all  $i$ . These results indicate that tabu search in the JSP can be viewed as a diffusion process with a central restoring force – the probability of moving closer to (farther from) the nearest optimal solution is proportional (inversely proportional) to the current distance from the nearest optimal solution. The impact of short-term memory is also evident in that the probability of continuing to move along the current gradient is very strong and exceeds 0.5 independently of  $i$  for nearly all problem instances.

The search cost predicted by the dynamic cost model is defined as the mean number of iterations until an absorbing state (i.e., a state that is distance 0 from an optimum) is encountered. To validate the Markov model, the predicted  $\bar{c}$  is estimated, for our  $6 \times 4$ ,  $6 \times 6$ , and  $10 \times 10$  problem sets, by repeatedly simulating the Markov chain defined by  $D$ , the set of states  $S_{i,grad}$ , and the estimated transition probabilities  $P(S_{i,grad}|S_{j,grad})$ . For our  $6 \times 4$  and  $6 \times 6$  instances,

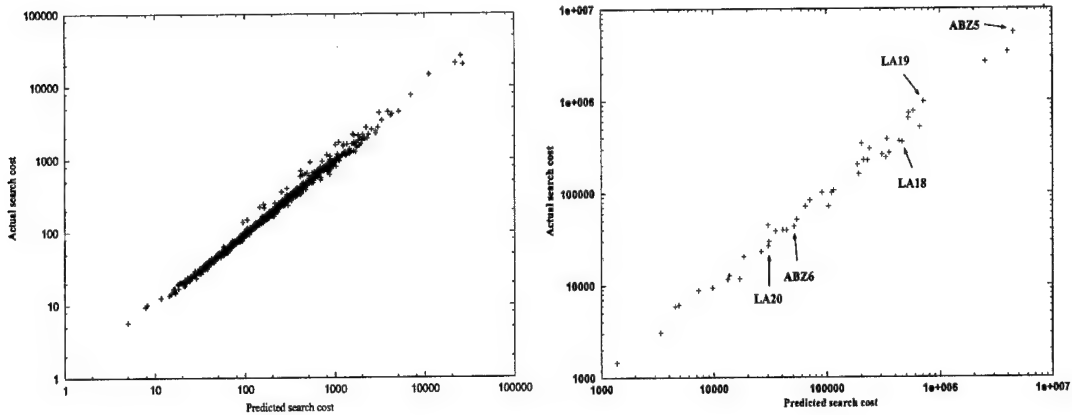


Figure 10: Scatter plots of the observed versus predicted mean cost ( $\bar{c}$ ) to locate an optimal solution under  $TS_{Taillard}$  for  $6 \times 6$  (left figure) and  $10 \times 10$  (right figure) random JSPs; the least-squares fit lines are superimposed.

$\log_{10}$ - $\log_{10}$  regression models of the predicted versus actual  $\bar{c}$  yielded  $r^2$  values of 0.9941 and 0.9939, respectively. Figure 10 shows scatter plots for  $6 \times 6$  and  $10 \times 10$  instances. For the  $10 \times 10$  instances, a  $\log_{10}$ - $\log_{10}$  regression model of the predicted versus actual  $\bar{c}$  yielded an  $r^2$  value of 0.9877. These results clearly demonstrate that the behavior of Taillard's algorithm can be modeled with high fidelity as a simple one-dimensional random walk. In contrast to both the  $\bar{d}_{lopt-opt}$  and  $\bar{d}_{tabu-opt}$  models, the Markov model appears scalable. Finally, we note that our Markov model is equally successful in accounting for the variability in the cost of locating suboptimal solutions to these same problem instances.

### 3 Air Force Satellite Control Network (AFSCN) Scheduling

The U.S. Air Force Satellite Control Network (AFSCN) is responsible for coordinating communications between numerous civilian and military organizations and more than 100 satellites. Space-ground communications are performed via 16 antennas located at 9 ground stations positioned around the globe. Customer organizations submit task requests to reserve an antenna at a ground station for a specific time period based on the windows of visibility between target satellites and available ground stations. Alternate time windows and ground stations may also be specified. As of 2000, over 500 task requests are received by the AFSCN scheduling center for a typical day. The communication antennas are over-subscribed in that many more task requests are received than can be accommodated. Currently, human scheduling experts construct an initial schedule that typically leaves about 120 conflicts representing task requests that are unsatisfiable. However, satellites are extremely expensive resources, and the AFSCN is expected to maximize satellite utilization; out-right rejection of task requests is not a viable option. Consequently, human schedulers must engage in a complex, time-consuming arbitration process between various organizations to eliminate all conflicts present in the initial schedule. The generic problem of scheduling task requests for communication antennas is referred to as the Satellite Range Scheduling Problem [24].

In reality, the processes and criteria that human schedulers use to develop conflict-free schedules are generally implicit and difficult to quantify, making automation extremely difficult. Instead, much of this research has focused on a single, although crucial, aspect of the problem: minimizing the number of conflicts in the initial schedule. Minimizing the number of conflicts up-front reduces (1) the workload of human schedulers, (2) communication with outside agencies, and (3) the time required to produce a conflict-free schedule. By emphasizing conflict minimization, we ignore many important aspects of an extremely complex real-world problem. For example, although a conflict-free schedule is produced at least 24 hours before execution, roughly one-third of the scheduled task requests might be modified because of real-time considerations such as high-priority emergency task requests or breakdowns at the ground station [24]. Conflict-minimization for Satellite Range Scheduling is emphasized as a core capability because the process lends itself to automation and because conflict minimization has been the focus of virtually all other researchers.

To put the application in context, Satellite Range Scheduling has been related to other well-known and studied scheduling problems. To do this, Satellite Range Scheduling (or, for brevity, "Range Scheduling") can be expressed in two forms: Single-Resource Range Scheduling (SRRS) and Multi-Resource Range Scheduling (MRRS). In Single-Resource Range Scheduling, there is only one resource (i.e., antenna) that is being scheduled; Multi-Resource Range Scheduling includes tasks that can potentially be scheduled on one of several alternative resources. A special case of Multi-Resource Range Scheduling occurs when the various tasks to be scheduled can be decomposed into multiple but independent single-resource range scheduling problems where there are no interactions between resources. Generally, however, alternatives are actually on different resources.

The problem of minimizing the number of unsatisfied task requests on a single antenna (SRRS) is equivalent to minimizing the number of late jobs on a single machine; this problem is known to be  $\mathcal{NP}$ -complete. This result has been used to formally establish that Range Scheduling is also  $\mathcal{NP}$ -complete [4]. Nevertheless, finding optimal or near optimal solutions to many realistic, single-resource problem instances is relatively easy. For some reasonable size problems, *branch and bound* methods can be used to either find or confirm optimal solutions.

Both exact methods (e.g., branch and bound methods) and heuristic methods, which were taken from the single machine scheduling literature to address the problem of Single-Resource Range Scheduling, have been implemented and studied in our project. These methods have been compared to scheduling algorithms found in the satellite scheduling literature. The Range Scheduling specific methods were largely designed for Multi-Resource Scheduling, and in fact, the single machine scheduling heuristics generally out-perform the best algorithms specifically designed for Range Scheduling when compared on Single-Resource Range Scheduling problems.

The majority of the research has focused on the Multi-Resource Range Scheduling problem. MRRS is not a single, well defined problem, but rather an on-going application in a dynamic environment. Previous researchers at the Air Force Institute of Technology (AFIT) tested their algorithms on actual data from 1992 (the "AFIT benchmark"). An important issue is whether the results from those data hold for current application conditions. For example, in 1992, approximately 300 requests needed to be scheduled for a single day, compared to 500 requests per day in recent years. More requests for the same resources have a clear impact on problem difficulty, but the pattern of usage (e.g., distribution of tasks across resources) may also have

changed. Recent data were obtained for the application<sup>2</sup>, and algorithms were compared on both AFIT (from 1992) and current data to investigate whether the problems in the AFIT benchmarks are representative of the kinds of Range Scheduling problems that are currently encountered by AFSCN.

From our study, it appears that the AFIT benchmarks are simple in the sense that a heuristic can be used to quickly find the best known solutions to these problems. The heuristic splits the tasks to be scheduled into “low-altitude satellite” requests and “high-altitude satellite” requests. Because low-altitude satellite requests are highly constrained, these requests are scheduled first. A theorem and proof have been presented showing that when a set of low-altitude requests is restricted to specific time slots, a greedy scheduling method exists that is optimal. The proof allows tasks to be scheduled on multiple alternative resources as long as the resources have identical capabilities. These results suggest that further research using the AFIT benchmarks could be misleading because these problems do not represent the kinds of scheduling problems that are currently faced by AFSCN.

Multi-Resource Range Scheduling problems, including current Range Scheduling problems (based on 2003 data obtained from Schriever), exist where simple heuristics do not yield optimal results. For larger problem instances, the heuristic of splitting tasks into low and high-altitude requests is no longer a good strategy. Additionally, in contrast to results on the SRRS problem, results on these large problems are consistent with the findings of Parish [22]: the *Genitor* algorithm outperforms other heuristics on the larger problem instances. Thus, it appears that although the algorithms previously tested for the Satellite Range Scheduling problem do not excel on the SRRS problem, these algorithms do generalize to more modern versions on the MRRS problem.

### 3.1 Variants of Range Scheduling with Differing Complexity

While the general problem of Satellite Range Scheduling is  $\mathcal{NP}$ -complete, special subclasses of Range Scheduling are polynomial. Burrowbridge (1999) considers a simplified version of the single-resource range scheduling problem, where only low-altitude satellites are present, and the objective is to maximize the number of scheduled tasks. Due to the orbital dynamics of low-altitude satellites, the task requests in this problem have negligible *slack*; i.e., the window size is equal to the request duration. The well-known *greedy activity-selector* algorithm [11] is used to schedule the requests since it yields a solution with the maximal number of scheduled tasks.

Range Scheduling for low-altitude satellites has been proven to have polynomial time complexity even if multiple resources are options [4]. In particular, this occurs in the case of scheduling low-altitude satellite requests on one of the  $k$  antennas present at a particular ground station. For the proof, the  $k$  antennas must represent equivalent resources.

The greedy activity-selector algorithm for multiple resource problems was modified as follows: the algorithm still schedules the requests in increasing order of their due date, however, it specifies that each request is scheduled on the resource for which the idle time before its start time is minimum. Minimizing this idle time is critical to proving the optimality of the

<sup>2</sup>We wish to thank Brian Bayless and William Szary from Schriever Air Force Base for the data and the feedback on our interpretation of the data files.

greedy solution. This algorithm, *Greedy<sub>IS</sub>* (where *IS* stands for Interval Scheduling), is optimal for scheduling the low-altitude requests. The problem of scheduling the low-altitude requests is equivalent to an interval scheduling problem with  $k$  identical machines (for more on interval scheduling, see Bar-Noy et al.[3], Spieksma [27], Arkin et al.[1]). It has been proven that for the interval scheduling problem, the extension of the greedy activity-selector algorithm is optimal; the proofs are based on the equivalence of the interval scheduling problem to the  $k$ -colorability of an interval graph [9].

### 3.2 Algorithms for the Satellite Range Scheduling Problem

Various heuristic algorithms have been applied to both Single-Resource Range Scheduling and Multi-Resource Range Scheduling. Algorithms for simpler problems are reviewed first; then algorithms for the more realistic Multi-Resource Range Scheduling are presented. The algorithms have been tested using the AFIT benchmarks, data from a problem generator, and actual data from Schriever Air Force Base from spring 2002 to spring 2003.

**Branch-and-Bound Algorithm for Single-Resource Range Scheduling** starts by computing a lower bound on the number of late tasks  $v$ . Branch-and-bound is then applied to the decision problem of finding a schedule with  $v$  late tasks. If no such schedule can be found,  $v$  is incremented, and the process is repeated. When solving the decision problem, at each node in the search tree, the branching scheme selects an unscheduled task and attempts to schedule the task. The choice of the task to be scheduled is made based on a heuristic that prefers small tasks with large time windows over large tasks with tight time windows. Dominance properties and constraint propagation are applied; then the feasibility of the new one-machine schedule is checked. If the schedule is infeasible, the algorithm backtracks, and the task is considered late. Our version is derived from the Baptistie et al. [2] algorithm. Our experiments show this algorithm can solve only relatively small single-resource problems and cannot be used for AFSCN problems.

**Constructive Heuristic Algorithms** begin with an empty schedule and iteratively add jobs to the schedule using local, myopic decision rules. These heuristics can generate solutions to even large problem instances in subsecond CPU time, but because they typically employ no backtracking, the resulting solutions are generally suboptimal. Constructive search algorithms were developed for SRRS based on texture-based [6] and slack-based [26] constraint-based scheduling heuristics. It was found that texture-based heuristics are effective when the total number of task requests is small (e.g.,  $n \leq 100$ ) for SRRS, but straightforward extensions of constraint-based technologies for multiple resources with alternatives (MRRS) did not appear to be effective. These results were somewhat surprising. Several researchers in Artificial Intelligence and several industrial companies focus on the use of constructive methods. Our research indicates, however, that the constraints for Multi-Resource Range Scheduling problems are such that the cascade of task interactions cause the number of interactions to grow exponentially.

**Local Search and Tabu Search** begin with one or more complete solutions; search proceeds via successive modifications to a series of complete solution(s). All local search algorithms for

this project encode solutions using a permutation  $\pi$  of the  $n$  task request IDs (i.e.,  $[1..n]$ ). A key component of any local search algorithm is the move operator. Because little problem-specific knowledge is available for Range Scheduling, we have selected a move operator known as the *shift* operator because it has been successfully applied to a number of well-known scheduling problems [30]. Given its relatively large neighborhood size, the shift operator is used in conjunction with next-descent hill-climbing. The neighbors of the current solution are examined in a random order, and the first neighbor with either a lower or equal fitness (i.e., number of bumps) is accepted. Search is initiated from a random permutation and terminates when a prespecified number of solution evaluations is exceeded.

Straightforward implementations of Tabu search [15] for Multi-Resource Range Scheduling were also developed, but the resulting algorithms were not competitive. The size of the local search neighborhood using shift and other well-known problem-independent move operators is of order  $O(500^2)$ . With such a large neighborhood, tabu search and other forms of neighborhood local search are impractical. Methods for reducing the neighborhood size were explored, but performance was extremely poor.

**The *Genitor* Genetic Algorithm** Previous studies of Range Scheduling by AFIT researchers indicate that the *Genitor* genetic algorithm [13] provides superior overall performance [22]. The version of the *Genitor* used here was originally developed for a warehouse scheduling application [42,44], but it has also been applied to problems such as job shop scheduling [33].

As in local search, solutions are encoded as permutations of the task request IDs. Like all genetic algorithms, *Genitor* maintains a population of solutions. In each step of the algorithm, a pair of parent solutions is selected, and a crossover operator is used to generate a single child solution, which then replaces the worst solution in the population. The result is a form of elitism, in which the best individual produced during the search is always maintained in the population. Selection of parent solutions is based on the rank of their fitness, relative to other solutions in the population. A linear bias is used such that individuals that are above the median fitness have a rank-fitness greater than one and those below the median fitness have a rank-fitness of less than one [43].

Typically, genetic algorithms encode solutions using bit-strings, which enable the use of "standard" crossover operators such as one-point and two-point crossover [16]. Because solutions in *Genitor* are encoded as permutations, a special crossover operator is required to ensure that the recombination of two parent permutations results in a child that (1) inherits good characteristics of both parents and (2) is still a permutation of the  $n$  task request IDs. Following Parish (1994), the current implementation uses Syswerda's (relative) order crossover operator, which preserves the relative order of the elements in the parent solutions in the child solution. Syswerda's crossover operator has been successfully applied in a variety of scheduling applications [28,34,29].

**Schedule Builder** is used to generate solutions from a permutation of request IDs. The schedule builder considers task requests in the order that they appear in  $\pi$ . For SRRS, the schedule builder attempts to schedule the task request within the specified time window. For the MRRS, each task request is assigned to the first available resource (from its list of alternatives)

Customer Type	Predictability	Fract. of Reqs. for Each Request Type			
		HS	PD	PC	Mu
Operations and Maintenance	0.9	0.85	0.0	0.1	0.05
Image Intelligence	0.3	0.0	0.5	0.5	0.0
Signal Intelligence	0.7	0.05	0.1	0.85	0.0

Table 1: Customer types.

and at the earliest possible starting time. If the request cannot be scheduled on any of the alternative resources, it is dropped from the schedule (i.e., bumped). The evaluation of a schedule is then defined as the total number of requests that are scheduled (for maximization) or inversely, the number of requests bumped from the schedule (for minimizing). The schedule builder translates the solutions of the local search and *Genitor* algorithms.

### 3.3 The Problem Generator

A good problem generator was necessary to carefully evaluate many algorithms across a wide range of scheduling conditions. A generator was needed because there is a very limited amount of real data available for the AFSCN scheduling domain and to provide experimental control. The problem generator was developed based on the characteristics of the AFSCN application in general and the features of the SRRS problems, the AFIT benchmark problems, and the recent data, specifically. The AFSCN schedules task requests on a per-day basis. Parameters model different types of requests encountered in the real-world satellite scheduling problem, such as downloading data from a satellite, transmitting information or commands from a ground station to a satellite, and checking the health and status of a satellite. The requests are classified into four possible types:

- State of health (HS): short (5-15 minutes), flexible, common
- Maneuver (Mu), for example changing the orbit of a satellite: longer (20-60 minutes), inflexible, rare
- Payload download (PD), for example downloading data from the satellite: long (10-30 minutes), more flexible, common
- Payload commanding (PC): variable length (5-60 minutes), flexible, common

This taxonomy is based on the fact that requests of a certain type share characteristics of duration, flexibility (the size of the time window versus the duration of the request) and frequency<sup>3</sup>.

As a second feature, the new generator introduces models for customer behavior. Three customer types are defined (see Table 1), based on the fact that, for example, some customers will mostly generate health and status requests or payload command requests. For each customer, the number of requests to be generated is determined as a fraction of the total number

<sup>3</sup>For now, periodic requests are not modeled and therefore do not use the frequency information.

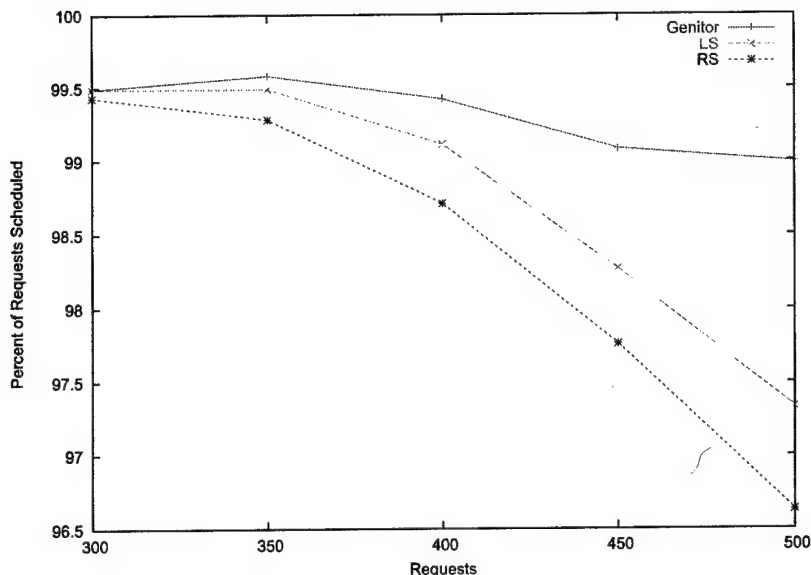


Figure 11: This figure illustrates how the *Genitor* algorithm scales compared to local search and a random sampling based method as problem size increases from 300 requests to 500 requests.

of requests. Customer patterns specify a time window midpoint, a resource, and the type of request. Customer patterns model the previous behavior of a customer (which could have been collected from past schedules). A database of customer patterns is used to produce the requests. For each customer type, the fraction of the requests generated for each request type and the *Predictability* are defined. The *Predictability* for each customer represents the fraction of requests that will be generated for this customer using customer patterns from the database. The rest of the requests corresponding to each customer will be randomly generated. The duration and window size for each request are determined using the parameters associated with the request type. The process of generating the requests in the new generator is described in the following publication [4].

This new generator facilitates testing scalability of algorithms and focuses efforts on those algorithms that can handle more realistic problem scenarios, both in terms of size and complexity. Figure 11 shows how the *Genitor* algorithm scales compared to local search and a random sampling based method as problem size increases from 300 requests to 500 request. These results suggest that *Genitor* scales better.

### 3.4 Multi-Resource Range Scheduling with Alternatives: Algorithm Analysis and the AFIT Benchmark Problems

Heuristics for Satellite Range Scheduling have previously been evaluated using only the seven problem instances in the AFIT benchmark. Although the set includes both high and low-altitude satellite requests along with alternatives, the low-altitude requests in these problems can be scheduled only at one ground station (by assigning it to one of the antennas present

Day	<i>Genitor</i>			Hill Climbing			Random Sampling			Greedy	MIP
	Min	Mean	Stdev	Min	Mean	Stdev	Min	Mean	Stdev		
1	8	8.6	0.49	15	18.16	2.54	21	22.7	0.87	20	10
2	4	4	0	6	10.96	2.04	11	13.83	1.08	19	6
3	3	3.03	0.18	11	15.4	2.73	16	17.76	0.77	24	7
4	2	2.06	0.25	12	17.43	2.76	16	20.20	1.29	20	7
5	4	4.1	0.3	12	16.16	1.78	15	17.86	1.16	18	6
6	6	6.03	0.18	15	18.16	2.05	19	20.73	0.94	27	7
7	6	6	0	10	14.1	2.53	16	16.96	0.66	22	6

Table 2: Performance of *Genitor*, hill climbing, and random sampling on the AFIT benchmark problems, in terms of the best and mean number of bumped requests. All statistics are taken over 30 independent runs. The results of running a greedy heuristic (*Greedy<sub>DP</sub>*) are also included. The last column reports the performance of Schalck’s MIP algorithm.

at that ground station). The number of requests to be scheduled for the seven problems are 322, 302, 300, 316, 305, 298, and 297, respectively. Since 1992, the number of requests received during a typical day has increased substantially (to more than 500 each day in recent years), while the resources have remained more or less constant.

To confirm the reported results from Parish’s study [22], *Genitor* was run on each of the seven problems in the benchmark, using the same parameters. Random sampling and hill-climbing were also run on each AFIT problem, with a limit of 8000 evaluations per run. Each algorithm performed a total of 30 independent runs on each problem. The results are summarized in Table 2. Included in the table are the results obtained by Schalck using Mixed Integer Programming (MIP) [24]. As previously reported, *Genitor* yields the best overall performance. A greedy heuristic method produced the worst performance (it is often outperformed by random sampling). Scheduling the primary requests in a greedy fashion drastically narrows the opportunities of scheduling the rest of the requests using the alternative resources.

### 3.4.1 Explaining the Performance on the Benchmarks

To exploit the differences in scheduling slack and the number of alternatives between low and high-altitude requests, a new greedy heuristic (which we call the “split heuristic”) was designed to first schedule all the low-altitude requests in the order given by the permutation, followed by the high-altitude requests. Experiments have shown that: (1) for more than 90% of the best-known schedules found by *Genitor*, the split heuristic does not increase the number of conflicts in the schedule, and (2) the split heuristic typically produces good (and often best-known) schedules.

From our analysis, it appears that *Genitor* learns to schedule the low-altitude requests before the high-altitude requests, leading to the strong overall performance. By separating the requests from the permutations produced by *Genitor* into low and high-altitude requests, the evaluation of more than 80% of the schedules remains unchanged.

As a followup, it was proven that *the low altitude requests are scheduled optimally by Genitor*. Since the split heuristic divides the low and high-altitude requests, the low-altitude requests (with no slack and no alternative windows for scheduling) correspond to multiple instances of

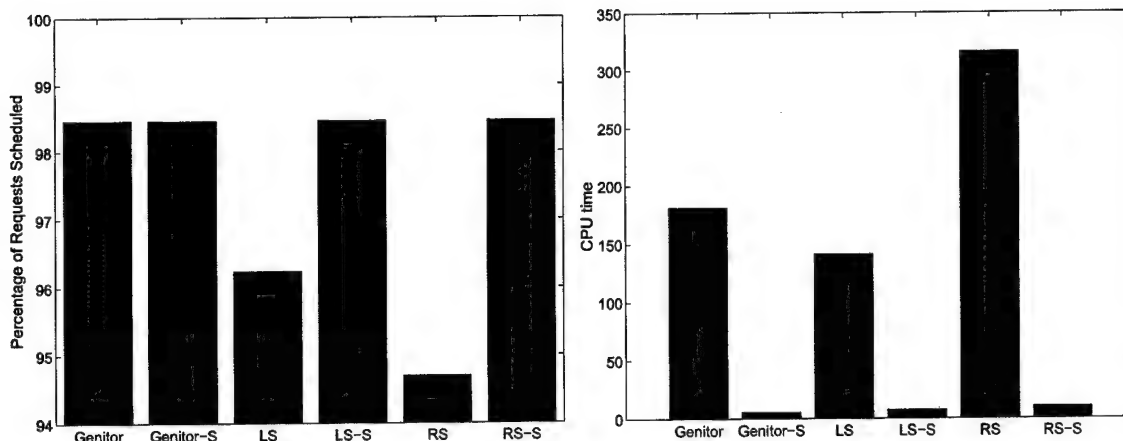


Figure 12: Algorithm performance for the seven AFIT benchmark problems.

SRRS problems with no slack and no time windows; therefore, these requests can be optimally scheduled when considered in isolation.

This result motivated the construction of a new version of Gooley's algorithm (which was developed for a M.S. thesis at AFIT) that is *provably guaranteed* to be as good as or better than Gooley's original algorithm. Gooley's algorithm uses Mixed Integer Programming (MIP) to schedule the low-altitude requests. It defines complex heuristics to obtain a permutation order of the high-altitude requests and to implement a schedule builder based on this permutation. His schedule builder inserts the high-altitude requests into the schedule (already containing the low-altitude requests) and then moves around requests in an attempt to schedule the unscheduled requests. An improved version of the heuristics used by Gooley's algorithm to schedule the high-altitude requests was also implemented.

An analysis of the influence of the initial permutation and of the schedule builder on Gooley's algorithm performance showed that our schedule builder results in fewer conflicts than Gooley's schedule builder when applied to the same permutation (the initial permutation built by Gooley's algorithm). Also, given one of the permutations evaluated by our schedule builder as the best solution obtained for that problem (smallest number of conflicts), Gooley's schedule builder results in more conflicts.

### 3.5 Generalizing the Satellite Scheduling Results: A Preliminary Study

To generalize the results obtained for Satellite Range Scheduling (SRS), a similar oversubscribed scheduling problem needed to be identified. The Management of the Missions of the Earth Observation Satellites <sup>4</sup> is a challenge problem put forth by the French Operations Research Society. The problem data are publicly available, as are the solutions found by other researchers.

<sup>4</sup>Data and problem description are available at <http://www.prism.uvrsq.fr/vdc/ROADEF/CHALLENGES/2003/challenge2003-en.html>.

For this new application, observation requests are collected from the customers. An observation request refers to some specified area on the Earth surface. The request is transformed into a set of strips covering the area. For each strip, two directions of acquisition are possible; one needs to be selected. A large number of requests is received for each day; typically they cannot be all satisfied. Setup times between pairs of acquisitions are specified. Also stereo requests are present (pairs of requests that both need to be scheduled, using the same direction of acquisitions). A solution specifies a feasible sequence of images to be acquired such that customer satisfaction is maximized. The objective function is the sum of the gains associated with the complete or partial acquisition of a request.

The availability of the new application allows us to investigate the influence of various problem features present in oversubscribed scheduling on algorithm performance, posing the question: Do the results obtained for SRS generalize for this similar problem? A simple greedy heuristic was designed and implemented for this problem. Also implemented was a local search algorithm that has been used in previous research for this problem [19]. A new objective function was implemented for Genitor.

As with the AFSCN application, Genitor outperformed the greedy heuristics and the local search algorithm. These results are preliminary; better results have been reported for this problem but the public posting does not list what algorithms produced those results. Preliminary results have also been obtained using a different objective function: minimizing the idle time (all the requests are scheduled on a single satellite).

### 3.6 Java Interface to AFSCN Algorithms

To facilitate examining solutions and obtaining feedback from actual schedulers, a Java GUI was built. The GUI allows the user to load in data from real or generated problems and to select an algorithm for scheduling. The resulting solution is displayed in an interface that lists each resource (antenna at a ground station) as a separate line. Optionally, bumped tasks can be displayed in red below the scheduled jobs so that the overlaps can be examined. Characteristics of scheduled and bumped tasks can be examined and changed (by clicking on the task number) if the user wishes. For example, if a task could be fit in if its start time were increased by a minute, the user could reset the start time, run the scheduler again and observe any improvement.

Figure 13 shows the interface; the primary display is zoomed in to show the tasks that have been scheduled on seven of the antennae. Times are listed below each line of jobs. Pull down menus at the top allow the user to change the algorithms being used, to load in other files, examine the set of bumped jobs, or change aspects of the display. The data were taken from actual requests from one day during spring 2003.

The interface also allows the user to examine alternative schedules or to manually fix portions of a schedule. For example, priority is not a factor considered while scheduling; the user can identify what is, in his opinion, a high priority job and force it to be placed in the schedule. When the scheduling algorithm is rerun, it will treat that assignment as given and schedule around it.

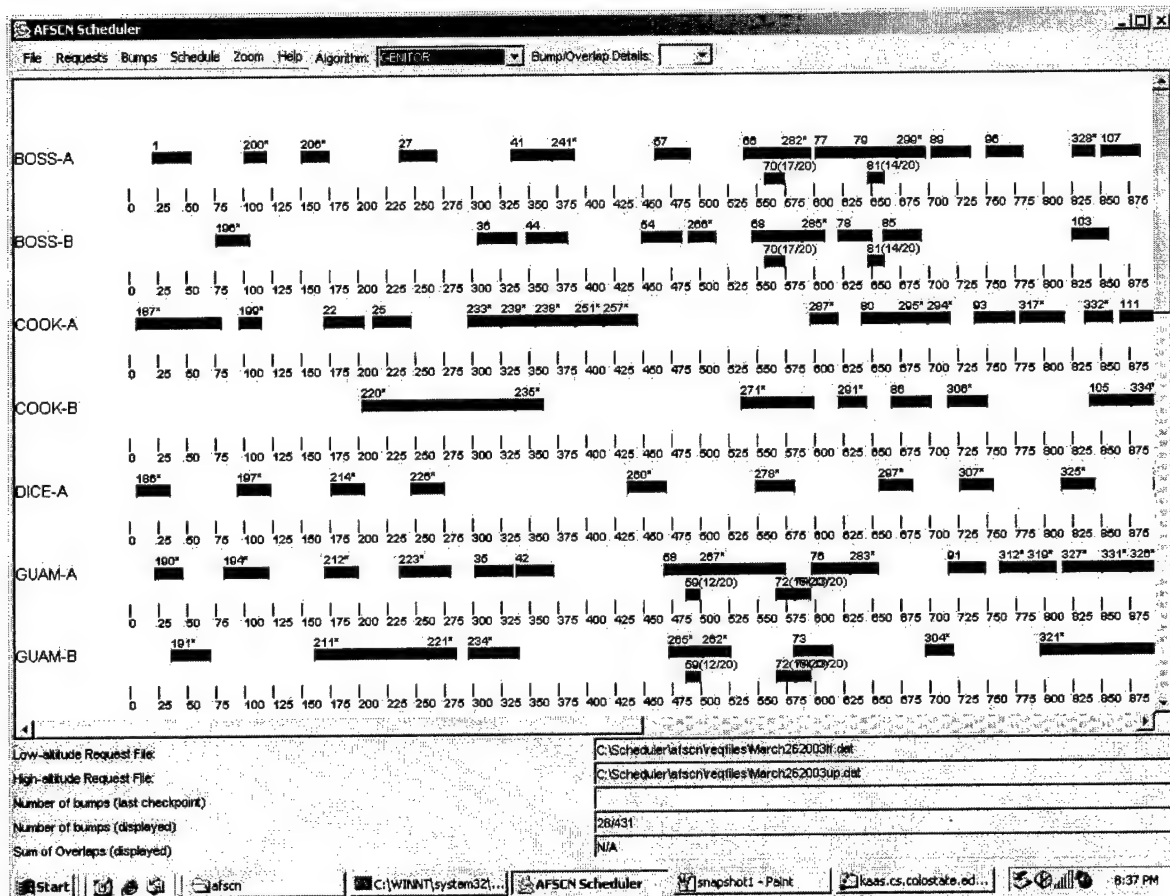


Figure 13: Java GUI for generating and examining AFSCN schedules. .

### 3.7 Conclusions about AFSCN Range Scheduling

The Satellite Range Scheduling Problem is an important real-world problem that impacts the use of expensive and limited resources. The study of this application has been structured as a progression. The progression started with a simple version of the problem, where only one resource was present. Next a version of the problem studied at AFIT was introduced. Then recent data was examined. Finally, for planning and experimental control purposes, a new problem generator was built that generalizes features found in the initial two versions of the problem studied and introduces new realistic features.

Although optimal algorithms have been developed for a restricted set of the SRRS problem, in general SRRS is NP-hard. It has been proven that no polynomial time algorithm exists for MRRS (unless the complexity class  $P = NP$ ). Additionally, the simple heuristics that proved so effective in the SRRS do not scale well to MRRS, especially for realistically sized problems.

Parish's earlier results with *Genitor* on AFIT's 1992 benchmarks have been shown to scale-up to the more realistic contemporary problems. It was also shown that *Genitor* performed so

well on the AFIT problems because it effectively learned to split the problems into low and high altitude requests, scheduling them separately. In fact, the seven problems in the AFIT benchmark are trivial to solve when a simple heuristic based on this separation is used.

However, when applied to more realistic problems, the split heuristic results in poor-quality solutions. Consequently, *Genitor* must be learning a different strategy for solving the contemporary problems. Experiments show that the *Genitor* algorithm yields better results compared to every other scheduling system we have tested on the real data set from Schriever.

## 4 Accomplishments/New Findings

This section lists the major results of our project organized by the two primary efforts. Our research on static models for JSP was awarded the PLANET (European network of excellence in AI planning) prize for research excellence in 2001. Darrell Whitley gave a keynote address about our work at *Fifth International Conference on Adaptive Computing in Design and Manufacturing (ACDM 2002)* in Exeter, England in April 2002.

### 4.1 Job Shop Scheduling Modeling and Studies

- Several studies of scheduling and planning have shown that algorithm results on a set of benchmark problems do not necessarily generalize: either to a larger set of synthetic problems or to more realistic problems. Additionally, the assumptions of comparative evaluation need to be carefully considered when making claims of generality.
- The search space features known to influence the cost of local search in SAT, specifically the number of optimal solutions ( $|optsols|$ ) and the mean distance between random solutions and the nearest optimal solution ( $d_{lopt-opt}$ ), also influence the cost of locating optimal solutions using tabu search in the JSP. Further, the *strength* of the influence of these two factors is nearly identical in both problems. As in SAT, we find that  $d_{lopt-opt}$  has a much stronger influence than  $|optsols|$  on search cost in the JSP and ultimately accounts for a significant proportion of the variance in search cost observed for a set of identically sized problem instances. This result was somewhat unexpected given the differences between the search spaces and local search algorithms of the JSP and SAT.
- For JSPs with moderate to large backbones, the correlation between backbone size and the number of optimal solutions is extremely high. As a direct consequence, for these problems, backbone size provides no more information than the number of optimal solutions, and vice versa: one of the two factors is necessarily redundant. Given the recent surge of interest in the link between backbone size and problem difficulty, the strong one-to-one correspondence between these two factors was completely unanticipated.
- No interaction effect was found between the backbone size and  $d_{lopt-opt}$ . Further, descriptive cost models based on either multiple factors or interacting factors are no more accurate than the simple model based solely on  $d_{lopt-opt}$ .

- A simple extension of the  $d_{lopt-opt}$  descriptive cost model accounts for most of the variance in the cost of finding *suboptimal* solutions to the JSP. This extension is the first quantitative model of the cost of locating suboptimal solutions to any *NP*-complete problem, and provides an explanation for the existence of ‘cliffs’ in the cost of finding suboptimal solutions of varying quality.
- For some time, researchers have observed that ‘square’ JSPs are generally more difficult than ‘rectangular’ JSPs. This phenomenon is likely due to differences in the distribution of  $d_{lopt-opt}$  for the two problem types. For square JSPs, the proportion of problem instances with large values of  $d_{lopt-opt}$  is substantial, while most instances of rectangular JSPs have very small values of  $d_{lopt-opt}$ .
- The accuracy of the  $\bar{d}_{lopt-opt}$  model can be significantly improved by considering the set of solutions visited by Taillard’s algorithm *during* search.
- Taillard’s algorithm can be modeled with exceptionally high fidelity using a surprisingly simple Markov chain whose states represent both the current distance from the nearest optimal solution and the current search gradient, i.e., whether search is progressing toward or away from the nearest optimal solution. The Markov model accounts for nearly all of the variability in the cost required to locate optimal solutions to both small ( $6 \times 4$ ,  $6 \times 6$ ) and large ( $10 \times 10$ ) random JSPs. The model also provides insight into the exact conditions under which different initialization methods can be expected to improve performance.
- The relationship between the Markov and  $\bar{d}_{lopt-opt}$  models has been characterized, which enables us to account for why  $\bar{d}_{lopt-opt}$  is so highly correlated with search cost.

## 4.2 Air Force Satellite Scheduling

- The 1992 AFIT benchmarks for the AFSCN Satellite Range Scheduling problem are simple in the sense that a heuristic was developed that quickly finds the best-known solutions to these problems. However, for larger problems, such as those faced by AFSCN today, simple heuristic solutions are inadequate.
- Range Scheduling for low-altitude satellites has been proven to have polynomial time complexity even if multiple resources are options. However, Satellite Range Scheduling in the general case has been proven to be NP-hard.
- A problem generator was developed based on the characteristics of the AFSCN application. This problem generator has allowed us to show how problem difficulty increases as the number of requests that needs to be assigned also increases. The problem generator also allows the PIs to evaluate how algorithm performance scales on larger size problems.
- The *Genitor* genetic algorithm produces the best results for minimizing the number of conflicts, especially on large problems and actual data from Schriever.
- Gooley’s algorithm, which was developed at AFIT, does not appear to scale well to the size and structure of recent problems. A study of the influence of the initial permutation and of

the schedule builder on Gooley's algorithm performance showed that our schedule builder results in fewer conflicts than Gooley's schedule builder when applied to the permutation constructed as part of Gooley's algorithm.

- A Java interface was developed so that the system could be demonstrated for schedulers working at Schriever Air Force Base. The interface also provided insights into what schedules look like and how conflicts might be resolved in order to repair a conflicted scheduler.
- Several scheduling algorithms have been developed and evaluated for a similar oversubscribed scheduling problem, The Management of the Missions of the Earth Observation Satellites. In a preliminary study, the *Genitor* genetic algorithm outperformed greedy heuristics and a local search algorithm.

## 5 Executive Summary

### 5.1 Personnel

During the grant period, the following personnel were supported at the indicated level:

PIs:			
Adele Howe	3.69	months	
L. Darrell Whitley	3.27	months	
Research Assistants:			
Laura Barbulescu	22.16	full-time months (9 full, 26.33 half-time)	
Jean-Paul Watson	19.9	full-time months (9 full-time, 21.8 half-time)	

### 5.2 Publications

- L. Barbulescu, J.P. Watson, D.L. Whitley, A.E. Howe. "Scheduling Space-Ground Communications for the Air Force Satellite Control Network", accepted to *Journal of Scheduling*.
- J.P. Watson, A.E. Howe, L.D. Whitley. 2003. "An Analysis of Iterated Local Search for Job-Shop Scheduling", To appear in *Proceedings of the Fifth Metaheuristics International Conference (MIC 2003)*, September.
- J.C. Beck, J.P. Watson. 2003. "Adaptive Search Algorithms and Fitness-Distance Correlation". To appear in the *Proceedings of the Fifth Metaheuristics International Conference (MIC-2003)*, September.
- J.P. Watson, L.D. Whitley, A.E. Howe. 2003. "A Dynamic Model of Tabu Search for the Job-Shop Scheduling Problem", To appear in the *First Multidisciplinary International Conference on Scheduling*, August.
- J.P. Watson, J.C. Beck, A.E. Howe, L.D. Whitley. 2003. "Problem Difficulty for Tabu Search in Job-Shop Scheduling", In *Artificial Intelligence*, Vol. 143, No. 2, pp. 189-217, February.

- L. Barbulescu, A.E. Howe, J.P. Watson and L.D. Whitley. 2002. "Satellite Range Scheduling: A Comparison of Genetic, Heuristic and Local Search", In *Proceedings of The Seventh International Conference on Parallel Problem Solving from Nature(PPSNVII)*, Granada, Spain, September.
- L.D. Whitley, J.P. Watson, A. Howe and L. Barbulescu. 2002. "Testing, Evaluation and Performance of Optimization and Learning Systems", Keynote Address at *Fifth International Conference on Adaptive Computing in Design and Manufacturing (ACDM 2002)*, Exeter, England, April.
- A.E. Howe and E. Dahlman. 2002. "A Critical Assessment of Benchmark Comparison in Planning", In *Journal of Artificial Intelligence Research*, Vol. 17, pp. 1-33, July.
- J.P. Watson, L. Barbulescu, L.D. Whitley, A.E. Howe. 2002. "Contrasting Structured and Random Permutation Flow-Shop Scheduling Problems: Search Space Topology and Algorithm Performance", In *INFORMS Journal on Computing*, Vol. 14, No. 1, Spring.
- D. Whitley. 2001. An Overview of Evolutionary Algorithms. *Journal of Information and Software Technology*. 43:817-831.
- J.P. Watson, J.C. Beck, A.E. Howe, and L.D. Whitley. 2001. "Toward a Descriptive Model Of Local Search Cost in Job-Shop Scheduling", In *Proceedings of Sixth European Conference on Planning (ECP'01)*, Toledo, Spain, September.
- J.P. Watson, J.C. Beck, A.E. Howe and L.D. Whitley. 2001. "Toward a Descriptive Model of Local Search Cost in Job-Shop Scheduling", In *Working Notes of IJCAI-01 Workshop on Stochastic Search Algorithms*, Seattle, WA, August.
- J.P. Watson, A.E. Howe. 2000. "Focusing on the Individual - Why We Need New Methods for Characterizing Problem Difficulty", In *Working Notes of ECAI 2000 Workshop on Empirical Methods in Artificial Intelligence*, Berlin, Germany, August.

### 5.3 Graduate Theses

One of the graduate research assistants will be completing his Ph.D. during summer 2003 and starting work at Sandia National Laboratory in early August. His thesis addresses issues central to the grant. The document can be obtained from the author or from Colorado State University.

**"Empirical Modeling and Analysis of Local Search Algorithms for the Job-Shop Scheduling Problem"** Ph.D. Thesis by Jean-Paul Watson in Summer 2003. Abstract:

Local search algorithms are among the most effective approaches for locating high quality solutions to a wide range of combinatorial optimization problems. However, our theoretical understanding of these algorithms is very limited, leading to significant problems for both researchers and practitioners. Specifically, the lack of a theory of local search impedes the development of more effective algorithms, prevents

practitioners from identifying the algorithm most appropriate for a given problem, and allows widespread conjecture and misinformation regarding the benefits and/or drawbacks of particular algorithms. This thesis represents a significant step toward a theory of local search. Using empirical methods, we develop theoretical models of the behavior of four well-known local search algorithms: a random walk, tabu search, iterated local search and simulated annealing. The analysis proceeds in the context of the well-known job-shop scheduling problem, one of the most difficult NP-hard problems encountered in practice. The large volume of prior research on the job-shop scheduling problem provides a diverse range of available algorithms and problem instances, in addition to numerous empirical observations regarding local search algorithm behavior; the latter are used to validate our behavioral models.

We show that all four local search algorithms can be modeled with high fidelity using straightforward variations of a generalized one-dimensional Markov chain. The states in these models represent sets of solutions a given fixed distance from the nearest optimal solution. The transition probabilities in all of the models are remarkably similar, in that search is consistently biased toward solutions that are roughly equidistant from the nearest optimal solution and solutions that are maximally distant from the nearest optimal solution. Surprisingly, the qualitative form of the transition probabilities is simply due to the structure of the representation used to encode solutions: the binary hypercube. The models account for between 96% and 99% of the variability in the cost required to locate both optimal and sub-optimal solutions to a wide range of problem instances, and provide explanations for numerous phenomena related to problem difficulty for local search in the job-shop scheduling problem. In the course of our analysis, we also disprove many conjectures regarding the behavior and benefits of particular algorithms.

Our research indicates that despite their effectiveness, local search algorithms for the job-shop scheduling problem exhibit surprisingly simple run-time dynamics. Further, we observe minimal differences between the dynamical behavior of different algorithms. As expected given similar run-time dynamics, although contrary to numerous reports appearing in the literature, we also show that the performance of different algorithms is largely indistinguishable. Ultimately, our behavioral models serve to unify and provide explanations for a large body of observations regarding problem difficulty for local search in the job-shop scheduling problem, and identify new research areas of the development of more effective local search algorithms.

The second research assistant, Laura Barbulescu, successfully defended her thesis proposal during the grant period. She expects to complete her Ph.D. in 2004.

**“Oversubscribed Scheduling Problems”** Ph.D. Thesis Proposal by Laura Barbulescu in Fall 2002.

## 5.4 Other Products

**Software** We have developed a prototype user interface in Java for the AFSCN scheduling problem. The interface allows a user to inspect proposed schedules and make minor changes to them.

We implemented state-of-the-art scheduling algorithms based on their descriptions in the literature; the re-implementations are in C++ to be run under the Linux operating system. The algorithms for JSP are:

**Taillard's Tabu Search with N1 Move Operator** [14] is a local search algorithm that uses a dynamic tabu tenure (a variable length list of recently visited states that should be avoided). The N1 operator is designed for the JSP and guarantees that a path exists from an arbitrary solution to some optimal solution.

**TSAB Tabu Search with N5 Move Operator** is a straightforward implementation of tabu search in conjunction with periodic re-intensification around previously encountered high quality solutions. N5 is generally a better move operator than N1 for the JSP, but it does not provide the path guarantee. Our version is a modified version of the original described in [21].

**Simulated Annealing with N1 Move Operator** is a stochastic, next descent local search algorithm in which the likelihood of accepting the improving move increases as search progresses. Our version is based on [32].

**Simulated Annealing with N5 Move Operator** is a variant of the original algorithm that uses the N5 move operator.

The algorithms for AFSCN are:

**Random Sampling** creating a random permutation for a schedule.

**Local Search** using the shift and swap operators over permutations.

**Three Satellite Interchange** heuristic for moving tasks around in a schedule that interchanges three tasks [17]. This was developed for an earlier solution to the AFSCN problem.

**GreedyDP** extension of a one machine heuristic based on [12].

**Split Heuristic** heuristic developed in house for favoring low altitude satellites before high altitude satellites.

We were able to obtain public domain versions of the code for the following algorithms:

**Genitor** A genetic algorithm developed by Darrell Whitley's lab.

We will make our software available on request.

**Web Site** Our project web site is available at <http://www.cs.colostate.edu/sched/>. From that site, you can access publications, problem instances and generators from the project.

## References

- [1] Esther M. Arkin and Ellen B. Silverberg. Scheduling Jobs with Fixed Start and End Times. *Discrete Applied Mathematics*, 18:1–8, 1987.
- [2] Philippe Baptiste, Claude Le Pape, and Laurent Peridy. Global Constraints for Partial CSPs: A Case-Study of Resource and Due Date Constraints. In Michael Maher and Jean-Francois Puget, editors, *Principles and Practice of Constraint Programming - CP98*, pages 87–101. Springer, 1998.
- [3] Amotz Bar-Noy, Sudipto Guha, Joseph (Seffi) Naor, and Baruch Schieber. Approximating the Throughput of Multiple Machines in Real-Time Scheduling. *SIAM Journal on Computing*, 31(2):331–352, 2002.
- [4] L.V. Barbulescu, J.P. Watson, L. D. Whitley, and A. E Howe. Scheduling space-ground communications for the air force satellite control network. *Journal of Scheduling*, to appear, 2000.
- [5] J. E. Beasley. *OR-LIBRARY*. <http://www.ms.ic.ac.uk/info.html>, 1998.
- [6] J. C. Beck, A. J. Davenport, E. M. Sitarski, and M. S. Fox. Texture-based Heuristic for Scheduling Revisited. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, pages 241–248, Providence, RI, 1997. AAAI Press / MIT Press.
- [7] Jacek Blażewicz, Erwin Pesch, and Malgorzata Sterna. The disjunctive graph machine representation of the job shop scheduling problem. *European Journal of Operational Research*, 127:317–331, 2000.
- [8] Sarah Elizabeth Burrowbridge. Optimal Allocation of Satellite Network Resources. In *Masters Thesis*. Virginia Polytechnic Institute and State University, 1999.
- [9] Martin C. Carlisle and Errol L. Lloyd. On the k-coloring of Intervals. *Discrete Applied Mathematics*, 59:225–235, 1995.
- [10] David A. Clark, Jeremy Frank, Ian P. Gent, Ewan MacIntyre, Neven Tomov, and Toby Walsh. Local search and the number of solutions. In *Proceedings of the Second International Conference on Principles and Practices of Constraint Programming (CP-96)*, pages 119–133, 1996.
- [11] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. MIT press, Cambridge, MA, 1990.
- [12] S. Dauzère-Pérès. Minimizing late jobs in the general one machine scheduling problem. *European Journal of Operational Research*, pages 131–142, 1995.
- [13] Lawrence Davis. *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York, 1991.

- [14] Éric D. Taillard. Parallel taboo search technique for the jobshop scheduling problem. Technical Report ORWP 89/11, DMA, Ecole Polytechnique Fédérale de Lausanne, Lausanne, Switzerland, 1989.
- [15] Fred Glover and Manuel Laguna. *Tabu Search*. Kluwer Academic Publishers, Boston, MA, 1997.
- [16] David Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA, 1989.
- [17] T.D. Gooley. Automating the Satellite Range Scheduling Process. In *Masters Thesis*. Air Force Institute of Technology, 1993.
- [18] A.E. Howe and E. Dahlman. A critical assessment of benchmark comparison in planning. *Journal of Artificial Intelligence Research*, 17:1–33, July 2002.
- [19] M. Lemaitre, G. Verfaillie, F. Jouhaud, J.M. Lachiver, and N. Bataille. Selecting and scheduling observations of agile satellites. *Aerospace Science and Technology*, 6:367–381, 2002.
- [20] Dirk C. Mattfeld, Christian Bierwirth, and Herbert Kopfer. A search space analysis of the job shop scheduling problem. *Annals of Operations Research*, 86:441–453, 1999.
- [21] E. Nowicki and C. Smutnicki. A fast tabu search algorithm for the permutation flow-shop problem. *European Journal of Operations Research*, 91:160–175, 1996.
- [22] D.A. Parish. A Genetic Algorithm Approach to Automating Satellite Range Scheduling. In *Masters Thesis*. Air Force Institute of Technology, 1994.
- [23] Andrew J. Parkes. Clustering at the phase transition. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence (AAAI-97)*, pages 340–345, 1997.
- [24] S.M. Schalck. Automating Satellite Range Scheduling. In *Masters Thesis*. Air Force Institute of Technology, 1993.
- [25] Josh Singer, Ian P. Gent, and Alan Smaill. Backbone fragility and the local search cost peak. *Journal of Artificial Intelligence Research*, 12:235–270, 2000.
- [26] S. Smith and C.C. Cheng. Slack-based Heuristics for Constraint Satisfaction Problems. In *Proceedings of the Eleventh National Conference on Artificial Intelligence (AAAI-93)*, pages 139–144, Washington, DC, 1993. AAAI Press / MIT Press.
- [27] Frits C.R. Spiessma. On the Approximability of an Interval Scheduling Problem. *Journal of Scheduling*, 2:215–227, 1999.
- [28] Gilbert Syswerda. Schedule Optimization Using Genetic Algorithms. In Lawrence Davis, editor, *Handbook of Genetic Algorithms*, chapter 21. Van Nostrand Reinhold, NY, 1991.

- [29] Gilbert Syswerda and Jeff Palmucci. The Application of Genetic Algorithms to Resource Scheduling. In L. Booker and R. Belew, editors, *Proc. of the 4th Int'l. Conf. on GAs*. Morgan Kaufmann, 1991.
- [30] E. Taillard. Some efficient heuristic methods for the flow shop sequencing problem. *European Journal of Operations Research*, 47:65–74, 1990.
- [31] Eric D. Taillard. Parallel taboo search techniques for the job shop scheduling problem. *ORSA Journal on Computing*, 6(2):108–117, 1994.
- [32] P.J.M van Laarhoven, E.H.L. Aarts, and J.K. Lenstra. Job shop scheduling by simulated annealing. *Operations Research*, 40(1):113–125, 1992.
- [33] M. Vazquez and D. Whitley. A Comparison of Genetic Algorithms for the Static Job Shop Scheduling Problem. In Schoenauer, Deb, Rudolph, Lutton, Merele, and Schwefel, editors, *Parallel Problem Solving from Nature*, 6, pages 303–312. Springer, 2000.
- [34] J. P. Watson, S. Rana, D. Whitley, and A. Howe. The Impact of Approximate Evaluation on the Performance of Search Algorithms for Warehouse Scheduling. *Journal of Scheduling*, 2(2):79–98, 1999.
- [35] Jean-Paul Watson. *Empirical Modeling and Analysis of Local Search Algorithms for the Job-Shop Scheduling Problem*. Computer science department, Colorado State University, Fort Collins, CO, Fall 2003.
- [36] J.P. Watson, L. Barbulescu, L.D. Whitley, and A.E. Howe. Contrasting structured and random permutation flow-shop scheduling problems: Search space topology and algorithm performance. *INFORMS Journal on Computing*, 14(1), Spring 2002.
- [37] J.P. Watson, J.C. Beck, A.E. Howe, and L.D. Whitley. Toward a descriptive model of local search cost in job-shop scheduling. In *Proceedings of Sixth European Conference on Planning (ECP'01)*, Toledo, Spain, September 2001.
- [38] J.P. Watson, J.C. Beck, A.E. Howe, and L.D. Whitley. Problem difficulty for tabu search in job-shop scheduling. *Artificial Intelligence*, 143(2):189–217, February 2003.
- [39] J.P. Watson and A.E. Howe. Focusing on the individual - why we need new methods for characterizing problem difficulty. In *Working Notes of ECAI 2000 Workshop on Empirical Methods in Artificial Intelligence*, Berlin, Germany, August 2000.
- [40] J.P. Watson, A.E. Howe, and L.D. Whitley. An analysis of iterated local search for job-shop scheduling. In *Proceedings of the Fifth Metaheuristics International Conference (MIC 2003)*, Japan, September 2003.
- [41] J.P. Watson, L.D. Whitley, and A.E. Howe. A dynamic model of tabu search for the job-shop scheduling problem. In *First Multidisciplinary International Conference on Scheduling*, England, August 2003.

- [42] Darrell Whitley, Timothy Starkweather, and D'ann Fuquay. Scheduling Problems and Traveling Salesmen: The Genetic Edge Recombination Operator. In J. D. Schaffer, editor, *Proc. of the 3rd Int'l. Conf. on GAs*. Morgan Kaufmann, 1989.
- [43] L. Darrell Whitley. The GENITOR Algorithm and Selective Pressure: Why Rank Based Allocation of Reproductive Trials is Best. In J. D. Schaffer, editor, *Proc. of the 3rd Int'l. Conf. on GAs*, pages 116–121. Morgan Kaufmann, 1989.
- [44] L. Darrell Whitley, Timothy Starkweather, and Daniel Shaner. The Traveling Salesman and Sequence Scheduling: Quality Solutions Using Genetic Edge Recombination. In Lawrence Davis, editor, *Handbook of Genetic Algorithms*, chapter 22, pages 350–372. Van Nostrand Reinhold, 1991.
- [45] L.D. Whitley, J.P. Watson, A. Howe, and L. Barbulescu. Testing, evaluation and performance of optimization and learning systems. In *Adaptive Computing in Design and Manufacturing*, 2002. Keynote Address.